

Distance Matrix Methods

Anders Gorm Pedersen

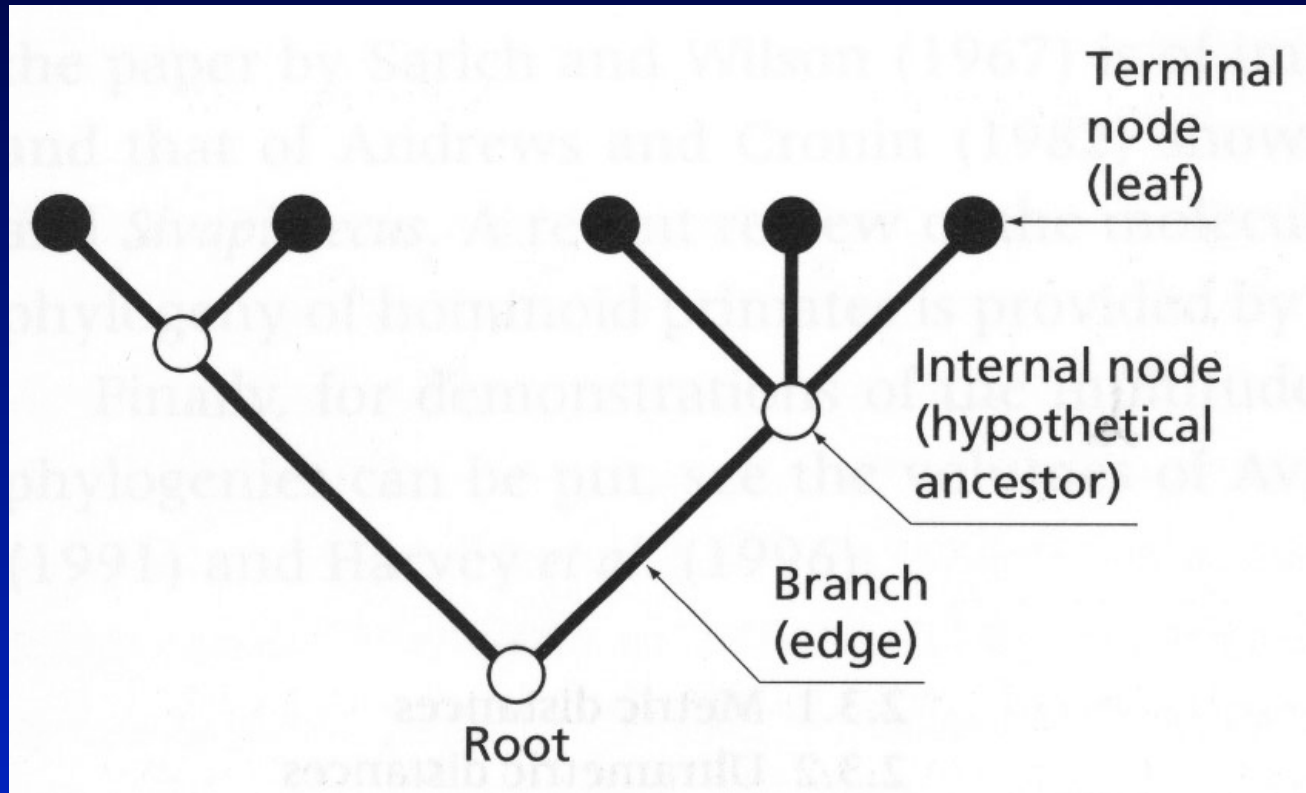
Molecular Evolution Group

Center for Biological Sequence Analysis

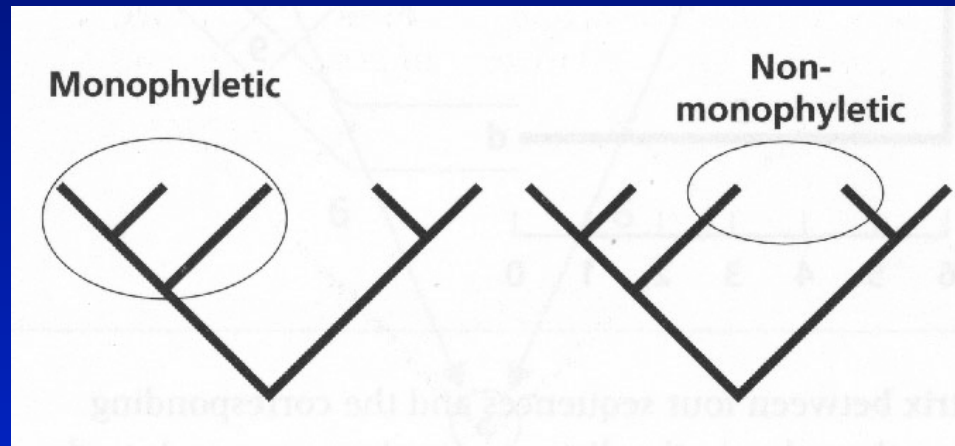
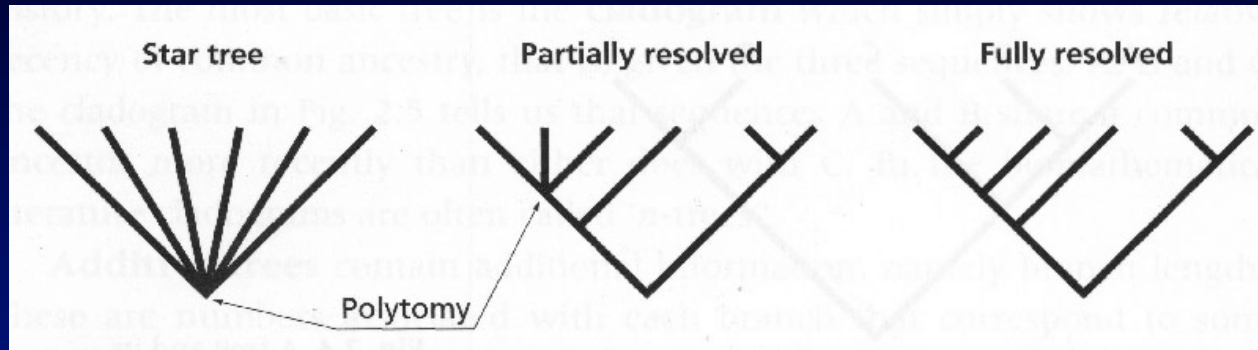
Technical University of Denmark

gorm@cbs.dtu.dk

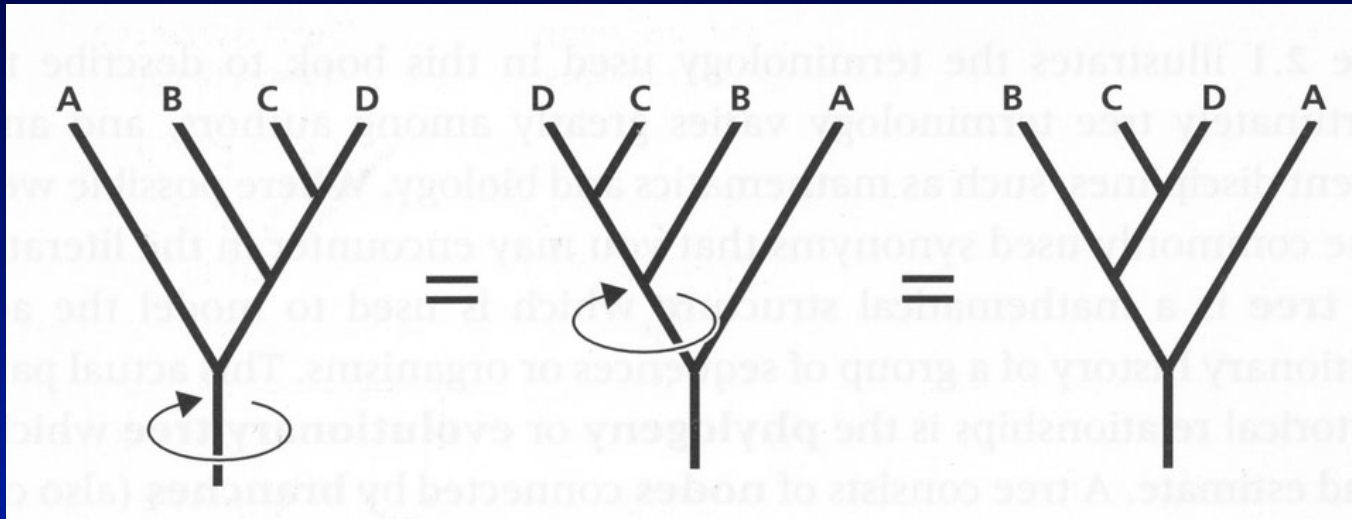
Trees: terminology



Trees: terminology

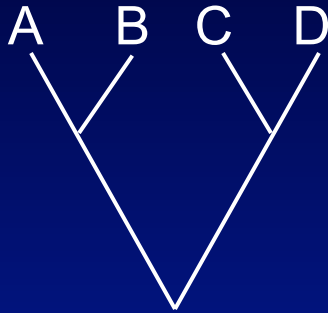


Trees: representations



Three different representations of the same tree

Trees: representation in computer files



((A , B) , (C , D));

Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Newick format: named for seafood restaurant where standard was decided upon



Trees: representation in computer files



Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Trees: representation in computer files



Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Trees: representation in computer files



Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

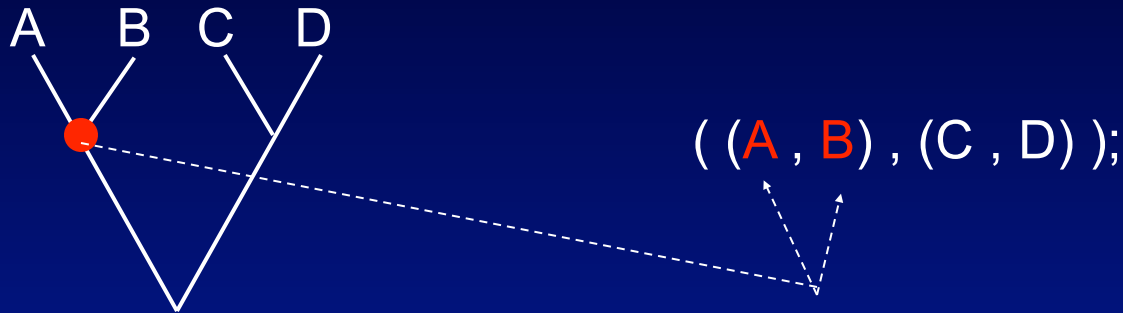
Trees: representation in computer files



Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

Trees: representation in computer files



Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

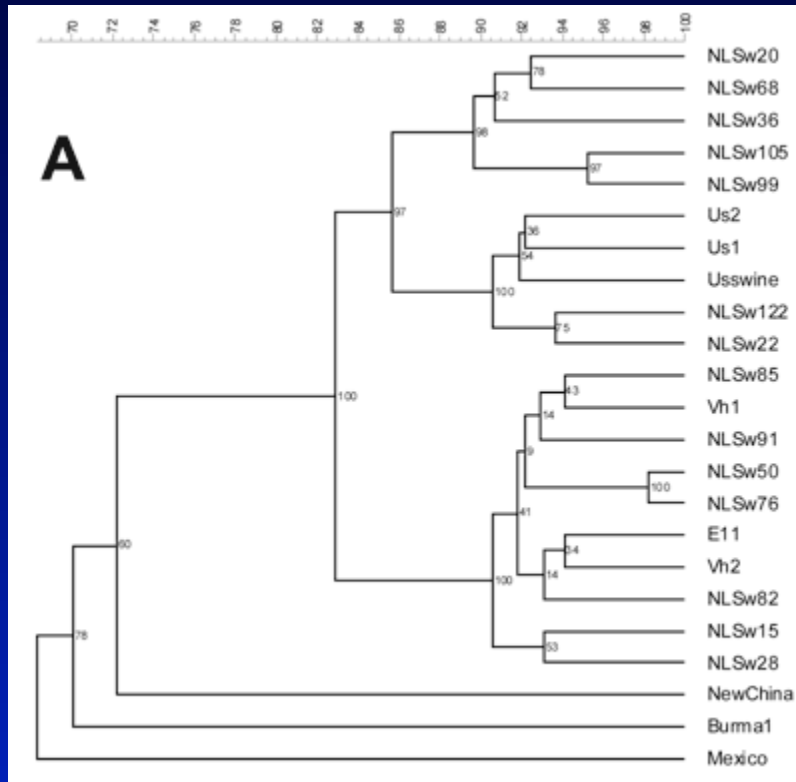
Trees: representation in computer files



Newick format:

- Leafs: represented by taxon name
- Internal nodes: represented by pair of matching parentheses
- Descendants of internal node given as comma-delimited list.
- Tree string terminated by semicolon

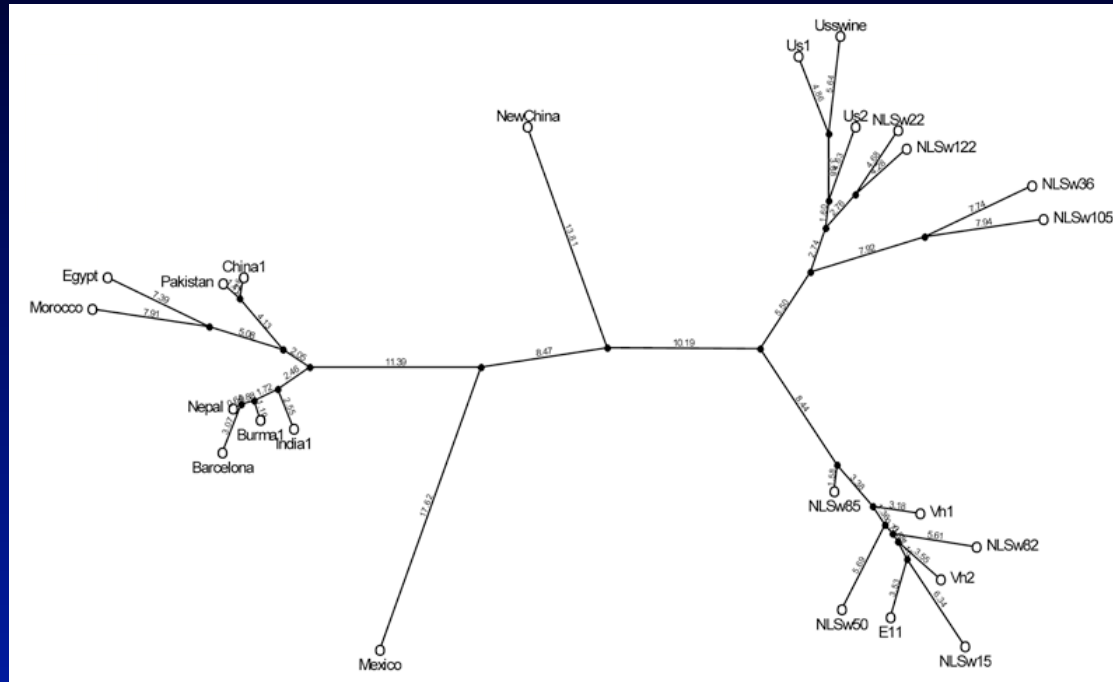
Trees: rooted vs. unrooted



- A rooted tree has a single node (the root) that represents a point in time that is earlier than any other node in the tree.
- A rooted tree has directionality (nodes can be ordered in terms of “earlier” or “later”).
- In the rooted tree, distance between two nodes is represented along the time-axis only (the second axis just helps spread out the leafs)

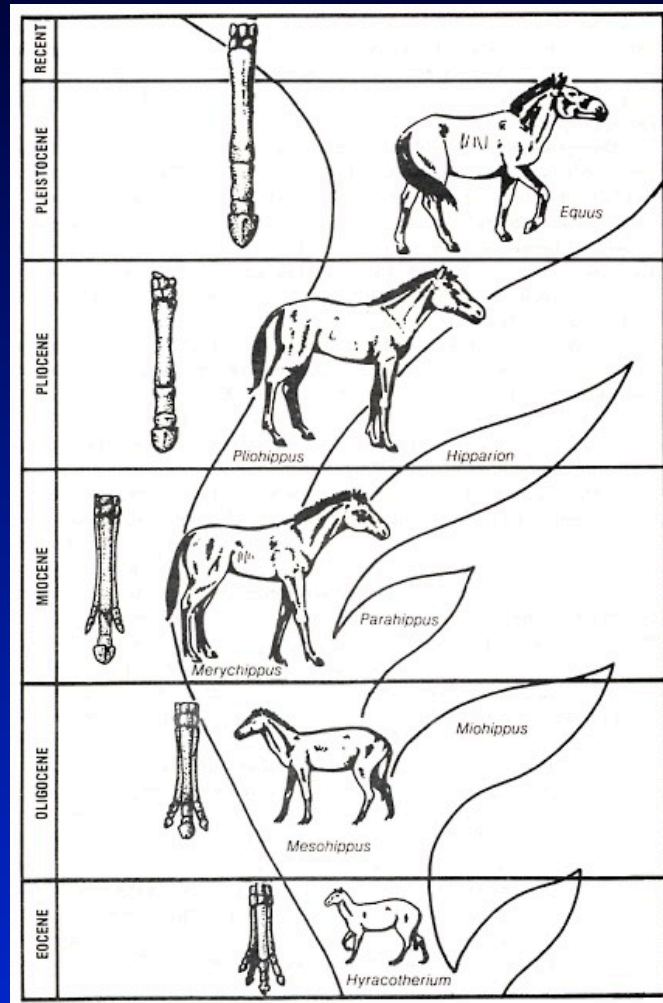
Early  Late

Trees: rooted vs. unrooted

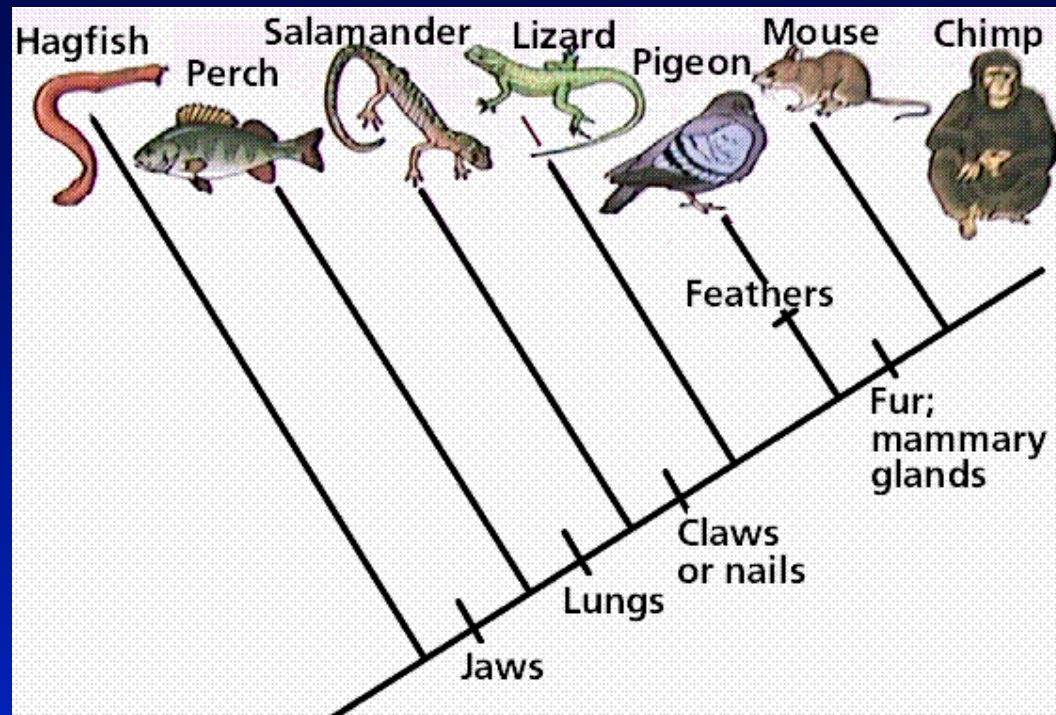


- In unrooted trees there is no directionality: we do not know if a node is earlier or later than another node
- Distance along branches directly represents node distance

Reconstructing a tree using non-contemporaneous data



Reconstructing a tree using present-day data



Data: molecular phylogeny

- DNA sequences
 - genomic DNA
 - mitochondrial DNA
 - chloroplast DNA
- Protein sequences
- Restriction site polymorphisms
- DNA/DNA hybridization
- Immunological cross-reaction

Morphology vs. molecular data



African white-backed vulture
(old world vulture)



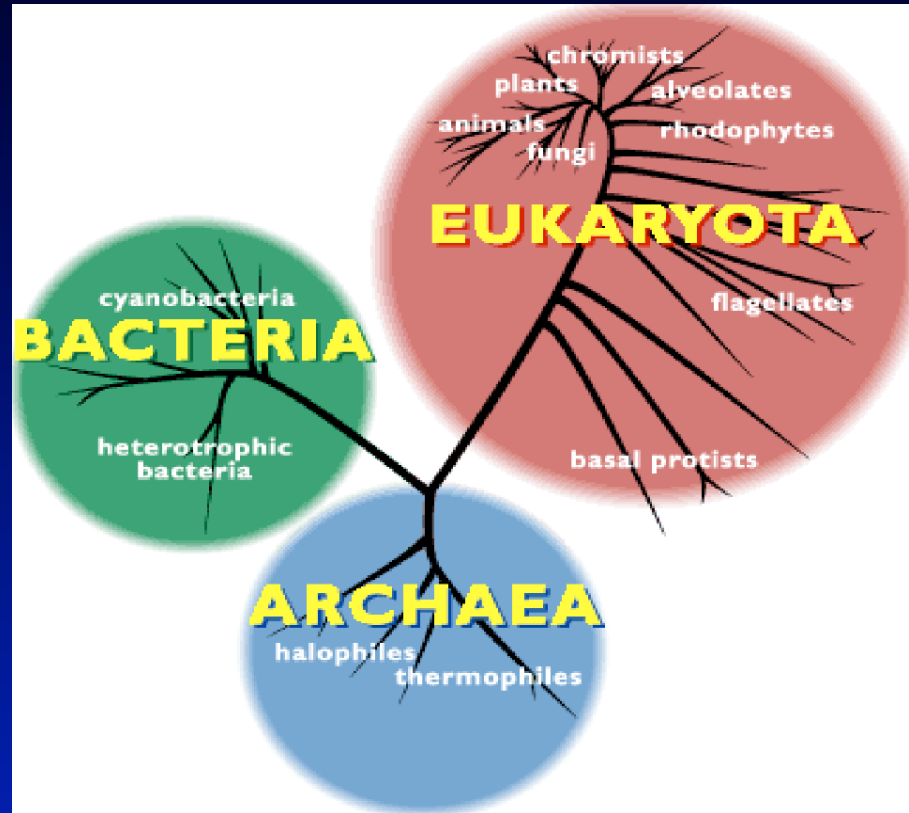
Andean condor
(new world vulture)

New and old world vultures seem to be closely related based on **morphology**.

Molecular data indicates that old world vultures are related to birds of prey (falcons, hawks, etc.) while new world vultures are more closely related to storks

Similar features presumably the result of convergent evolution

Molecular data: single-celled organisms



Molecular data useful for analyzing single-celled organisms (which have only few prominent morphological features).

Distance Matrix Methods

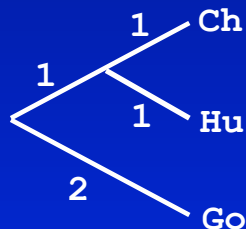
Gorilla : ACGT**CGTA**
 Human : ACGTTCCT
 Chimpanzee: ACGTT**TCG**

↓ ↓ ↓ ↓
 ↑ ↑

1. Construct multiple alignment of sequences

	Go	Hu	Ch
Go	-	4	4
Hu		-	2
Ch			-

2. Construct table listing all pairwise differences (distance matrix)

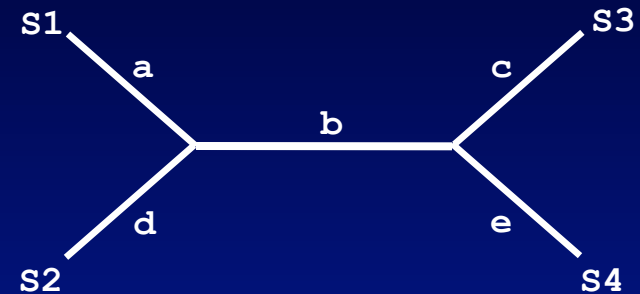


3. Construct tree from pairwise distances

Finding optimal branch lengths

	s_1	s_2	s_3	s_4
s_1	-	D_{12}	D_{13}	D_{14}
s_2		-	D_{23}	D_{24}
s_3			-	D_{34}
s_4				-

Observed distance



Distance along tree
(patristic distance)

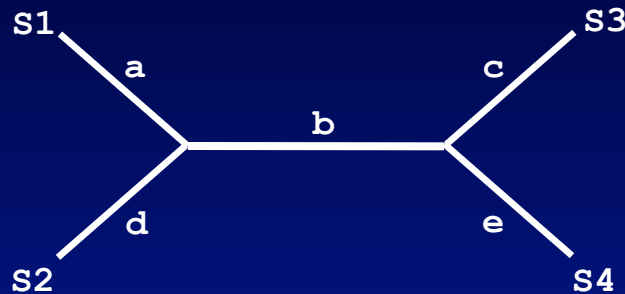
Goal:

$$\begin{aligned}
 D_{12} &\approx d_{12} = a + d \\
 D_{13} &\approx d_{13} = a + b + c \\
 D_{14} &\approx d_{14} = a + b + e \\
 D_{23} &\approx d_{23} = d + b + c \\
 D_{24} &\approx d_{24} = d + b + e \\
 D_{34} &\approx d_{34} = c + e
 \end{aligned}$$

Exercise (handout)

- Construct distance matrix (count different positions)
- Reconstruct tree and find best set of branch lengths

Optimal Branch Lengths for a Given Tree: Least Squares



Distance along tree

- Fit between given tree and observed distances can be expressed as “sum of squared differences”:

$$Q = \sum_{j>i} (D_{ij} - d_{ij})^2$$

Goal:

$D_{12} \approx d_{12} = a + d$
$D_{13} \approx d_{13} = a + b + c$
$D_{14} \approx d_{14} = a + b + e$
$D_{23} \approx d_{23} = d + b + c$
$D_{24} \approx d_{24} = d + b + e$
$D_{34} \approx d_{34} = c + e$

- Find branch lengths that minimize Q
- this is the optimal set of branch lengths for this tree.

Optimal Branch Lengths for a Given Tree: Least Squares Example

S1: TCCGAGTCGATCAGC
S2: ACCGAGTCGATCTGC
S3: AAGTACCCGTTGATC
S4: AAGTTGCCGTTTCAGG

Multiple alignment

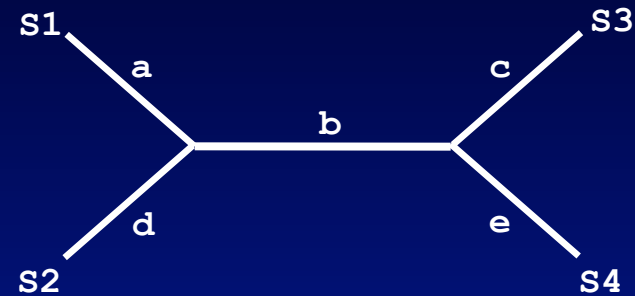
	S_1	S_2	S_3	S_4
S_1	–	2	9	8
S_2		–	9	8
S_3			–	5
S_4				–

Observed distance

Optimal Branch Lengths for a Given Tree: Least Squares Example

	s_1	s_2	s_3	s_4
s_1	-	2	9	8
s_2		-	9	8
s_3			-	5
s_4				-

Observed distance



Distance along tree

Goal: find branch lengths that minimize Q

$$\begin{aligned}
 Q &= \sum_{j>i} (D_{ij} - d_{ij})^2 \\
 &= (D_{12} - d_{12})^2 + (D_{13} - d_{13})^2 + \\
 &\quad (D_{14} - d_{14})^2 + (D_{23} - d_{23})^2 + \\
 &\quad (D_{24} - d_{24})^2 + (D_{34} - d_{34})^2
 \end{aligned}$$

Where:

$$\begin{aligned}
 d_{12} &= a + d \\
 d_{13} &= a + b + c \\
 d_{14} &= a + b + e \\
 d_{23} &= d + b + c \\
 d_{24} &= d + b + e \\
 d_{34} &= c + e
 \end{aligned}$$

Finding Optimal Branch Lengths

$$\frac{\partial Q}{\partial a} = 0$$

$$\frac{\partial Q}{\partial b} = 0$$

$$\frac{\partial Q}{\partial c} = 0$$

$$\frac{\partial Q}{\partial d} = 0$$

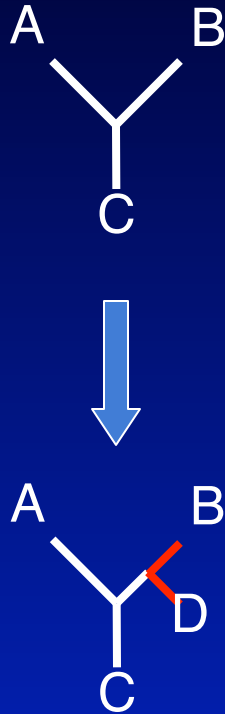
$$\frac{\partial Q}{\partial e} = 0$$

- System of n linear equations with n unknowns
- Can be solved for a, b, c, d, e

Least Squares Optimality Criterion

- Search through all (or many) tree topologies
- For each investigated tree, find best branch lengths using least squares criterion (solve N equations with N unknowns)
- Among all investigated trees, the best tree is the one with the smallest sum of squared errors.
- Least squares criterion used both for finding branch lengths on individual trees, and for finding best tree.

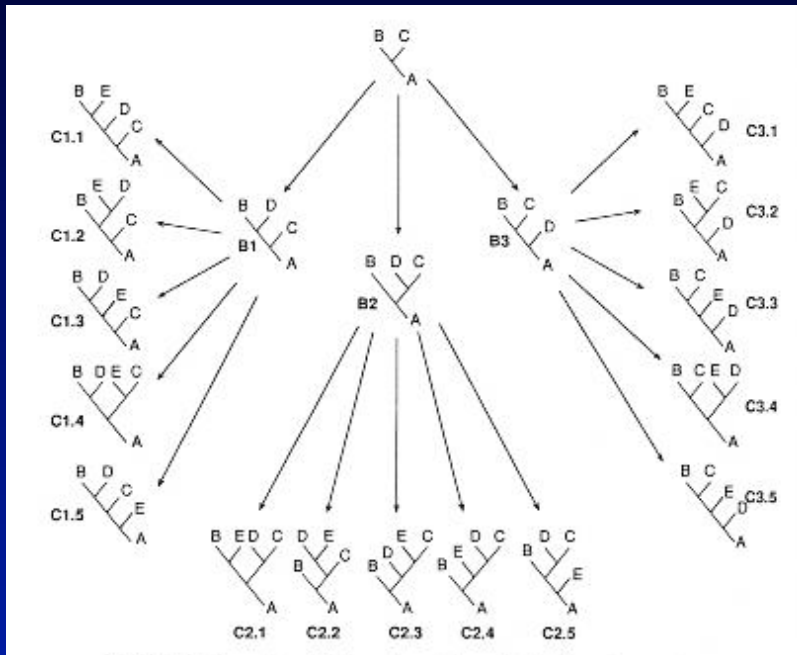
How many branches are there on an unrooted tree with x tips?



- There is only one way of constructing the first tree. This tree has 3 tips and 3 branches
- Each time an extra taxon is added, two branches are created.
- A tree with x tips will therefore have the following number of branches:

$$\begin{aligned}n_{\text{branches}} &= 3 + (x-3) \cdot 2 \\&= 3 + 2x - 6 \\&= 2x - 3\end{aligned}$$

How many unrooted trees are there?



- A tree with x tips has $2x-3$ branches
- For each tree with x tips, we can therefore construct $2x-3$ derived trees (which each have $x+1$ tips).

How many unrooted trees are there?

N_{tips}	N_{trees}	$N_{\text{branches}} = N_{\text{derived trees}}$
3	1	$2 \times 3 - 3 = 3$
4	1×3	$2 \times 4 - 3 = 5$
5	$1 \times 3 \times 5$	$2 \times 5 - 3 = 7$
6	$1 \times 3 \times 5 \times 7$	$2 \times 6 - 3 = 9$
7	$1 \times 3 \times 5 \times 7 \times 9$	$2 \times 7 - 3 = 11$
8	$1 \times 3 \times 5 \times 7 \times 9 \times 11$	$2 \times 8 - 3 = 13$
9	$1 \times 3 \times 5 \times 7 \times 9 \times 11 \times 13$...

$$N_{\text{trees with } n \text{ tips}} = \prod_{i=2}^{n-1} (2i - 3)$$

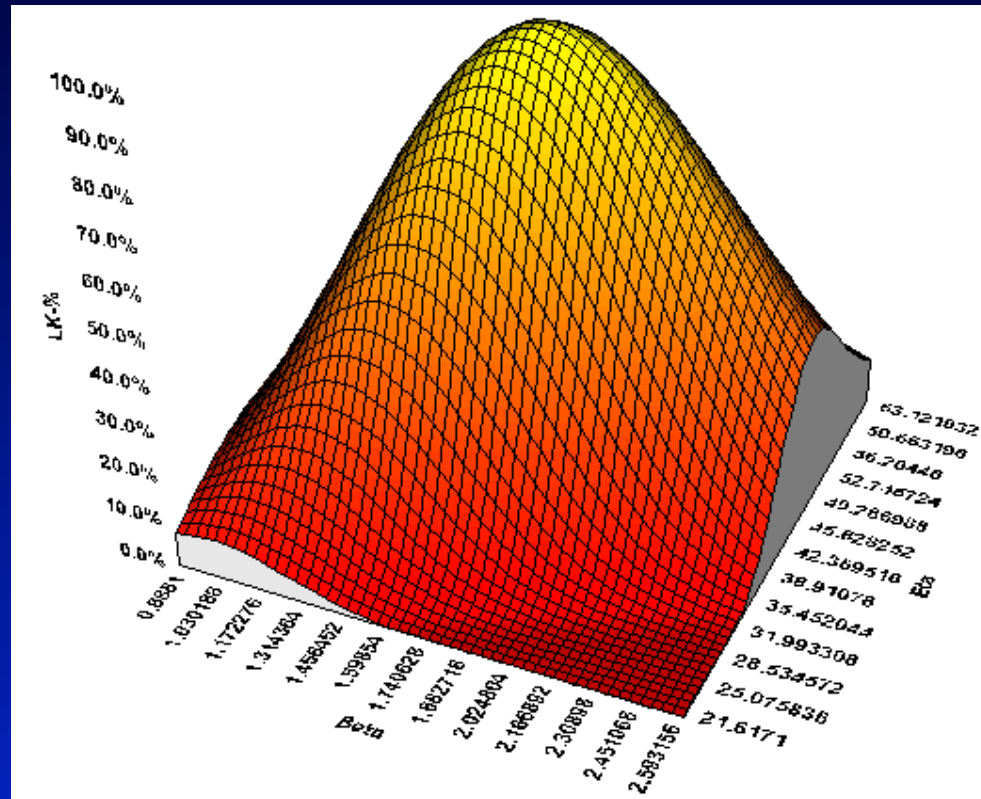
Exhaustive search impossible for large data sets

No. taxa	No. trees
3	1
4	3
5	15
6	105
7	945
8	10,395
9	135,135
10	2,027,025
11	34,459,425
12	654,729,075
13	13,749,310,575
14	316,234,143,225
15	7,905,853,580,625

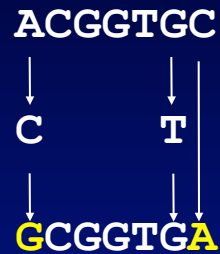
Heuristic search

1. Construct initial tree; determine sum of squares
2. Construct set of “neighboring trees” by making small rearrangements of initial tree; determine sum of squares for each neighbor
3. If any of the neighboring trees are better than the initial tree, then select it/them and use as starting point for new round of rearrangements. (Possibly several neighbors are equally good)
4. Repeat steps 2+3 until you have found a tree that is better than all of its neighbors.
5. This tree is a “local optimum” (not necessarily a global optimum!)

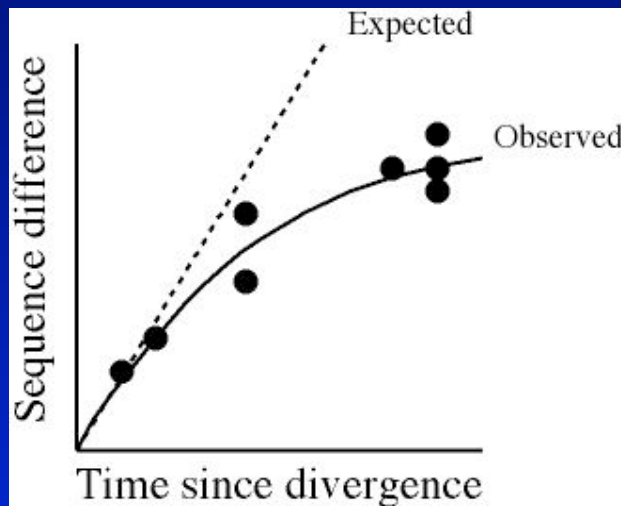
Heuristic search: hill-climbing



Superimposed Substitutions



- Actual number of evolutionary events: 5
- Observed number of differences: 2

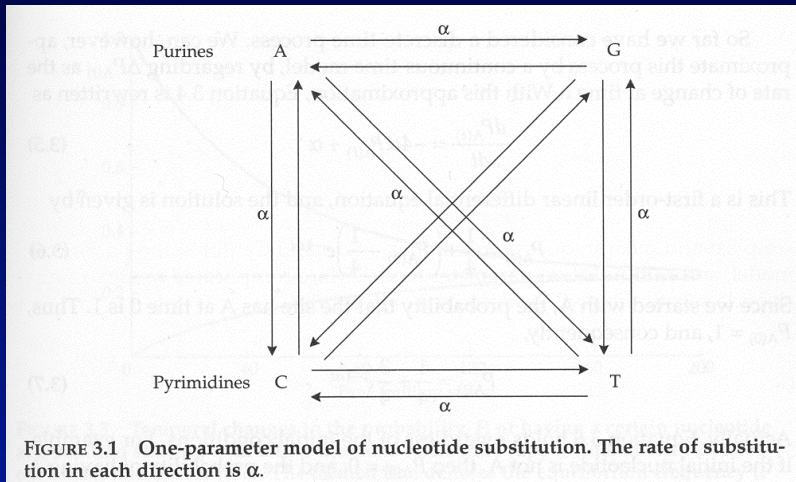


- Distance is (almost) always underestimated

Model-based correction for superimposed substitutions

- Goal: try to infer the real number of evolutionary events (the real distance) based on
 1. Observed data (sequence alignment)
 2. A model of how evolution occurs

Jukes and Cantor Model



- Four nucleotides assumed to be equally frequent ($f=0.25$)
- All 12 substitution rates assumed to be equal
- Under this model the corrected distance is:

$$D_{JC} = -0.75 \times \ln(1 - 1.33 \times D_{OBS})$$

- For instance:
 $D_{OBS}=0.43 \Rightarrow D_{JC}=0.64$

	A	C	G	T
A	-3α	α	α	α
C	α	-3α	α	α
G	α	α	-3α	α
T	α	α	α	-3α

Clustering Algorithms

- Starting point: Distance matrix
- Cluster the two nearest nodes:
 - Tree: connect pair of nodes to common ancestral node, compute branch lengths from ancestral node to both descendants
 - Distance matrix: replace the two joined nodes with the new (ancestral) node. Compute new distance matrix, by finding distance from new node to all other nodes
- Repeat until all nodes are linked in tree
- Results in only one tree, there is no measure of tree-goodness.

Neighbor Joining Algorithm

- For each tip compute $u_i = \sum_j D_{ij} / (n-2)$
(essentially the average distance to all other tips, except the denominator is n-2 instead of n-1)
- Find the pair of tips, i and j, where $D_{ij} - u_i - u_j$ is smallest
- Connect the tips i and j, forming a new ancestral node. The branch lengths from the ancestral node to i and j are:

$$v_i = 0.5 D_{ij} + 0.5 (u_i - u_j)$$

$$v_j = 0.5 D_{ij} + 0.5 (u_j - u_i)$$

- Update the distance matrix: Compute distance between new node and each remaining tip as follows:

$$D_{ij,k} = (D_{ik} + D_{jk} - D_{ij}) / 2$$

- Replace tips i and j by the new node which is now treated as a tip
- Repeat until only two nodes remain.

Neighbor Joining Algorithm

	A	B	C	D
A	-	17	21	27
B		-	12	18
C			-	14
D				-

Neighbor Joining Algorithm

	A	B	C	D
A	-	17	21	27
B		-	12	18
C			-	14
D				-

i	u_i
A	$(17+21+27) / 2 = 32.5$
B	$(17+12+18) / 2 = 23.5$
C	$(21+12+14) / 2 = 23.5$
D	$(27+18+14) / 2 = 29.5$

Neighbor Joining Algorithm

	A	B	C	D
A	-	17	21	27
B		-	12	18
C			-	14
D				-

i	u_i
A	$(17+21+27) / 2 = 32.5$
B	$(17+12+18) / 2 = 23.5$
C	$(21+12+14) / 2 = 23.5$
D	$(27+18+14) / 2 = 29.5$

	A	B	C	D
A	-	-39	-35	-35
B		-	-35	-35
C			-	-39
D				-

$$D_{ij} - u_i - u_j$$

Neighbor Joining Algorithm

	A	B	C	D
A	-	17	21	27
B		-	12	18
C			-	14
D				-

i	u_i
A	$(17+21+27) / 2 = 32.5$
B	$(17+12+18) / 2 = 23.5$
C	$(21+12+14) / 2 = 23.5$
D	$(27+18+14) / 2 = 29.5$

	A	B	C	D
A	-	-39	-35	-35
B		-	-35	-35
C			-	-39
D				-

$$D_{ij} - u_i - u_j$$

Neighbor Joining Algorithm

	A	B	C	D
A	-	17	21	27
B		-	12	18
C			-	14
D				-

i	u_i
A	$(17+21+27) / 2 = 32.5$
B	$(17+12+18) / 2 = 23.5$
C	$(21+12+14) / 2 = 23.5$
D	$(27+18+14) / 2 = 29.5$

	A	B	C	D
A	-	-39	-35	-35
B		-	-35	-35
C			-	-39
D				-

$$D_{ij} - u_i - u_j$$



Neighbor Joining Algorithm

	A	B	C	D
A	-	17	21	27
B		-	12	18
C			-	14
D				-

i	u_i
A	$(17+21+27) / 2 = 32.5$
B	$(17+12+18) / 2 = 23.5$
C	$(21+12+14) / 2 = 23.5$
D	$(27+18+14) / 2 = 29.5$

	A	B	C	D
A	-	-39	-35	-35
B		-	-35	-35
C			-	-39
D				-

$$D_{ij} - u_i - u_j$$



$$v_C = 0.5 \times 14 + 0.5 \times (23.5 - 29.5) = 4$$

$$v_D = 0.5 \times 14 + 0.5 \times (29.5 - 23.5) = 10$$

Neighbor Joining Algorithm

	A	B	C	D	X
A	-	17	21	27	
B		-	12	18	
C			-	14	
D				-	
X					-

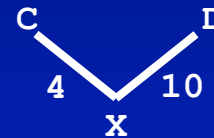


Neighbor Joining Algorithm

	A	B	C	D	X
A	-	17	21	27	
B		-	12	18	
C			-	14	
D				-	
X					-

$$\begin{aligned}
 D_{XA} &= (D_{CA} + D_{DA} - D_{CD}) / 2 \\
 &= (21 + 27 - 14) / 2 \\
 &= 17
 \end{aligned}$$

$$\begin{aligned}
 D_{XB} &= (D_{CB} + D_{DB} - D_{CD}) / 2 \\
 &= (12 + 18 - 14) / 2 \\
 &= 8
 \end{aligned}$$

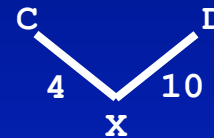


Neighbor Joining Algorithm

	A	B	C	D	X
A	-	17	21	27	17
B		-	12	18	8
C			-	14	
D				-	
X					-

$$\begin{aligned}
 D_{XA} &= (D_{CA} + D_{DA} - D_{CD}) / 2 \\
 &= (21 + 27 - 14) / 2 \\
 &= 17
 \end{aligned}$$

$$\begin{aligned}
 D_{XB} &= (D_{CB} + D_{DB} - D_{CD}) / 2 \\
 &= (12 + 18 - 14) / 2 \\
 &= 8
 \end{aligned}$$

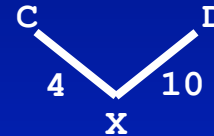


Neighbor Joining Algorithm

	A	B	X
A	-	17	17
B		-	8
X			-

$$\begin{aligned}
 D_{XA} &= (D_{CA} + D_{DA} - D_{CD}) / 2 \\
 &= (21 + 27 - 14) / 2 \\
 &= 17
 \end{aligned}$$

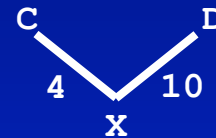
$$\begin{aligned}
 D_{XB} &= (D_{CB} + D_{DB} - D_{CD}) / 2 \\
 &= (12 + 18 - 14) / 2 \\
 &= 8
 \end{aligned}$$



Neighbor Joining Algorithm

	A	B	X
A	-	17	17
B		-	8
X			-

i	u_i
A	$(17+17) / 1 = 34$
B	$(17+8) / 1 = 25$
X	$(17+8) / 1 = 25$

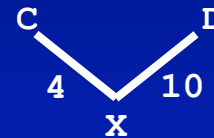


Neighbor Joining Algorithm

	A	B	X
A	-	17	17
B		-	8
X			-

i	u_i
A	$(17+17)/1 = 34$
B	$(17+8)/1 = 25$
X	$(17+8)/1 = 25$

	A	B	X
A	-	-42	-28
B		-	-28
X			-



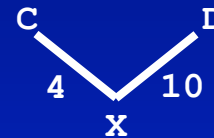
$$D_{ij} - u_i - u_j$$

Neighbor Joining Algorithm

	A	B	X
A	-	17	17
B		-	8
X			-

i	u_i
A	$(17+17)/1 = 34$
B	$(17+8)/1 = 25$
X	$(17+8)/1 = 25$

	A	B	X
A	-	-42	-28
B		-	-28
X			-



$$D_{ij} - u_i - u_j$$

Neighbor Joining Algorithm

	A	B	X
A	-	17	17
B		-	8
X			-

i	u_i
A	$(17+17)/1 = 34$
B	$(17+8)/1 = 25$
X	$(17+8)/1 = 25$

	A	B	X
A	-	-42	-28
B		-	-28
X			-

$$D_{ij} - u_i - u_j$$

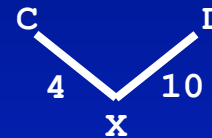


$$v_A = 0.5 \times 17 + 0.5 \times (34 - 25) = 13$$

$$v_D = 0.5 \times 17 + 0.5 \times (25 - 34) = 4$$

Neighbor Joining Algorithm

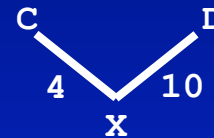
	A	B	X	Y
A	-	17	17	
B		-	8	
X			-	
Y				-



Neighbor Joining Algorithm

	A	B	X	Y
A	-	17	17	
B		-	8	
X			-	4
Y				-

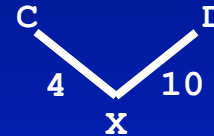
$$\begin{aligned}
 D_{YX} &= (D_{AX} + D_{BX} - D_{AB}) / 2 \\
 &= (17 + 8 - 17) / 2 \\
 &= 4
 \end{aligned}$$



Neighbor Joining Algorithm

	X	Y
X	-	4
Y		-

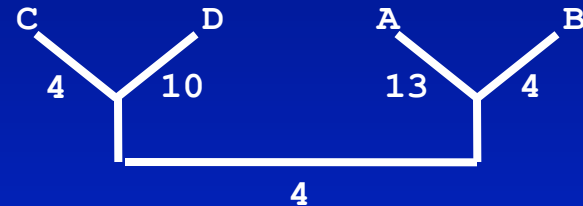
$$\begin{aligned}
 D_{YX} &= (D_{AX} + D_{BX} - D_{AB}) / 2 \\
 &= (17 + 8 - 17) / 2 \\
 &= 4
 \end{aligned}$$



Neighbor Joining Algorithm

	X	Y
X	-	4
Y		-

$$\begin{aligned}
 D_{YX} &= (D_{AX} + D_{BX} - D_{AB}) / 2 \\
 &= (17 + 8 - 17) / 2 \\
 &= 4
 \end{aligned}$$



Neighbor Joining Algorithm

	A	B	C	D
A	-	17	21	27
B		-	12	18
C			-	14
D				-

